Flashcards

File: Trees-annotated

Date: 25-07-2025

Total Cards: 10

Card 1/10

Q: What is the fundamental difference between a computer 'solving' a problem and 'executing' a solution, as stated by Laurent Gasser?

A: Computers do not inherently understand or devise solutions. They require a pre-defined, step-by-step solution (an algorithm) to be provided, which they then execute mechanically. 'Solving' implies understanding and devising a solution, while 'executing' implies following instructions.

Card 2/10

Q: Define a 'tree' data structure in terms of its components and relationships.

A: A tree is a hierarchical data structure consisting of nodes connected by edges. It has a root node, which has no parent, and subtrees, which are themselves trees. Nodes can have children, and nodes with the same parent are siblings.

Card 3/10

Q: Explain the difference between the 'height' and 'level' of a node in a tree.

A: The 'level' of a node is its distance from the root (root is at level 0). The 'height' of a node is the length of the longest path from that node to a leaf node. The 'height' of the tree is the height of the root node.

Card 4/10

Q: What distinguishes a 'strictly binary tree' from a 'full binary tree'?

A: A 'strictly binary tree' is a binary tree where every node has either 0 or 2 children. A 'full binary tree' is a binary tree where every node, except the leaf nodes, has exactly two children, and all leaf nodes are at the same level.

Card 5/10

Q: Describe the three common forms of expression notation (prefix, infix, postfix) and how they relate to the construction of an expression tree.

A: Prefix notation places the operator before its operands (e.g., + AB). Infix notation places the operator between its operands (e.g., A + B). Postfix notation places the operator after its operands (e.g., AB+). An expression tree can be constructed from any of these notations, with the operators as internal nodes and operands as leaf nodes. The order of operations is implicitly defined by the tree structure.

Card 6/10

Q: What is a 'balanced binary tree', and why is balance important?

A: A balanced binary tree is a binary tree where the heights of the left and right subtrees of every node differ by at most a certain constant (often 1). Balance is important because it ensures that operations like search, insertion, and deletion have a time complexity of O(log n), where n is the number of nodes. An unbalanced tree can degenerate into a linked list, resulting in O(n) complexity.

Card 7/10

Q: Explain the difference between 'Depth First Search' (DFS) and 'Breadth First Search' (BFS) in the context of binary tree traversal.

A: DFS explores as far as possible along each branch before backtracking. Common DFS traversals are Inorder, Preorder, and Postorder. BFS explores all the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. It uses a queue data structure.

Card 8/10

Q: What is a 'Binary Search Tree' (BST), and what property defines its structure?

A: A Binary Search Tree (BST) is a binary tree where for each node, the value of all nodes in its left subtree are less than the node's value, and the value of all nodes in its right subtree are greater than the node's value. This property allows for efficient searching, insertion, and deletion operations.

Card 9/10

Q: Describe the three cases to consider when deleting a node from a Binary Search Tree.

A: Case 1: The node to be deleted is a leaf node. Simply remove the node. Case 2: The node to be deleted has only one child. Replace the node with its child. Case 3: The node to be deleted has two children. Replace the node with its inorder successor (or predecessor) and then delete the inorder successor (or predecessor).

Card 10/10

Q: What are 'Heaps', and how do 'Max Heaps' and 'Min Heaps' differ?

A: A Heap is a specialized tree-based data structure that satisfies the heap property. In a 'Max Heap', the value of each node is greater than or equal to the value of its children. In a 'Min Heap', the value of each node is less than or equal to the value of its children. Heaps are often implemented as complete binary trees.